

PET ASCII OUTPUT PROGRAM FOR THE TIMEX/SINCLAIR ZX81

Introduction by Syncware News Technical Director, Fred Nachbaur

Here is another article by Carol F. Milazzo, describing a way of transmitting your ZX81 programs and data to other computers, such as the Commodore C64. As in her previous article (Audio Frequency Counter, SWN 2:5), Carol demonstrates some interesting and useful machine-code techniques which are readily applicable to other applications. One of the most fascinating aspects of the program (for me) is that it uses the ZX ROM's own "de-tokenizing" routines to send Sinclair BASIC in ASCII format. This is done by first printing (or listing) to the display file, then translating the contents of the display file to ASCII and shipping it out via a simple interface.

Ms. Milazzo's interface is designed specifically with the Commodore in mind. We are adding an alternate interface which provides an RS232-compatible output. You could then use your ZX81/TS1000 to send ASCII text to an RS232 terminal, printer, or other device. We'll leave the exact implementation up to you. Although parameters like baud rate and stop bits are elegantly catered to, others (like parity check bits, presence/absence of inverse characters or lower case) will have to be added if your outside device requires them.

Some of you may recall that we had intended to publish Bob Berch's article on driving RS232 printers with your ZX81. However, since Bob's implementation has been readily available from him for some time (and at a most reasonable price), and since it was considerably longer than we have had space for in this medium, we had to postpone it, and postpone it again. Ms. Milazzo's alternative is considerably shorter, and more easily extensible to other applications.

TS2068 CHALLENGES: Can any of you 2068'ers work out a similar way of shipping ASCII text via your EAR port? This one should be easy, since your character set is already coded in ASCII. How about a routine for transmitting ZX81 programs/data to the 2068, and vice versa, through the MIC and EAR ports? In this case, you might want to take a slightly different approach, and translate the tokens as well as the alphanumeric and symbol characters. Also, you'll have to figure out a way to emulate a tape signal (Hint: "borrow" the required code snippets from the 2068 EXROM, and from David Wood's book "Practical Guide to M/L on the TS1000").

Whatever your preference machine-wise, we hope you can find uses for the techniques presented here.

PET ASCII OUTPUT PROGRAM FOR THE TIMEX/SINCLAIR ZX81

Milazzo, Carol F., "PET ASCII Output Program for the Timex/Sinclair ZX81", *Syncware News*, Vol 3:3, January-February 1986. A Z80 machine language program to port ZX81 programs and

data to other computers.

INTRODUCTION

This article describes a project which may be useful for anyone desiring to use software for the ZX81, TS1000 or TS1500 on a different computer. The program version presented here is designed to convert Sinclair BASIC programs into ASCII text which can be sent from the Sinclair computer to the serial port of a Commodore C64 computer. The program can be used with other computers if the non-standard graphic characters are eliminated or appropriately changed. In addition to the program presented here, the user will require a terminal program that permits the receiving computer to download and save text, e.g. "Plus/Term" in Feb. 1985 Compute!, and a very simple hardware circuit to interface the computers.

The program described here consists of a portion in BASIC which is used to set the parameters, and to relocate the 327-byte Zilog Z80 machine language (ML) program which is in the first REM statement. Typically, 327 bytes can be reserved above RAMTOP by lowering it to 32441 (in 16K) using the following sequence: POKE 16388,185; POKE 16389,126; NEW; then LOAD "PET ASCII".

THE BASIC PROGRAM

Upon running the BASIC program, the prompt "RELOCATE?" will appear. Enter "N" or ENTER alone if you desire to run the ML program at its original location in the REM statement. The response "Y" will cause the computer to request "NEW LOCATION=" which should be answered with 32441 in this example. Lines 110-140 copy the ML program to the new location and lines 150-220 make the required adjustments to the addresses in the ML code. The computer will then print "OUTSCREEN = RAND USR 32568, OUTPROG = PRINT USR 32568". This information should be noted, as these are the commands required to send a display screen or a BASIC program listing as ASCII text.

The computer then requests, "BAUD RATE?", which can be selected up to 2400 baud. Then it prompts "STOP BITS?" to select the number of stop bits between characters. By experience, one stop bit is usually sufficient for data speeds up to 1200 baud. At least 2 stop bits should be used for higher speeds.

In lines 290-400, the program calculates the required constants and POKes them into the appropriate locations. The variable F in line 310 is set to the computer's clock frequency, nominally 3.25 MHz. Variable G in line 320 represents a fixed number of clock cycles in the bit length delay loop. (The "+10" adjusts the duration of the M/L commands at 40D3h and 40D4h, which only exist to allow a very fine adjustment in bit duration.) Then, "PRESS ENTER TO TRANSFER PROGRAM," and the ML routine is activated. (You can use this as a "test run" to make sure your system is working before transferring other programs.)

After the original program is NEWed, any desired BASIC program can be LOAded and transferred to a receiving computer by entering PRINT USR 32685, since the ML program is protected from the NEW command.

```
0 REM...(327 BYTES FOR ML)...
10 REM PET ASCII OUTPUT PROGRAM BY CAROL F. MILAZZO
20 PRINT "RELOCATE?"
30 INPUT A$
40 CLS
```

```

50 LET ORG=16514
60 LET DIS=0
70 IF A$<"Y" THEN GOTO230
80 PRINT "NEW LOCATION=";
90 INPUT LOC
100 PPINT LOC
110 LET DIS=LOC-ORG
120 FOR I=ORG TO ORG+326
130 POKE I+DIS,PEEK I
140 NEXT I
150 LET A$="166011661416619166261668416726167381674816753167881683216839"
160 FOR I=1 TO 56 STEP 5
170 LET A=VAL A$(I TO I+4)
180 LET AD=PEEK A+256*PEEK (A+1)+DIS
190 LET H=INT (AD/256)
200 POKE A+DIS+1,H
210 POKE A+DIS,AD-256*H
220 NEXT I
230 LET BH=16587+DIS
240 LET BL=16588+DIS
250 LET SB=16624+DIS
260 LET OS=16641+DIS
270 LET OP=16758+DIS
280 PRINT "OUTSCREEN = RAND USR ";OS,"OUTPROG = PRINT USR ";OP
290 PRINT "BAUD RATE? ";
300 INPUT R
310 LET F=3.25E6
320 LET G=93+10
330 LET M=INT (F/R+.5)
340 LET C=INT ((M-G+3310)/3339)
350 LET B=(M-G-3339*C+3323)/13
360 POKE BH,C
370 POKE BL,INT B
380 PRINT R,, "STOP BITS? "
390 INPUT S
400 POKE SB,S
410 PRINT S,, "PRESS ENTER TO TRANSFER PROGRAM"
420 INPUT A$
430 PRINT USR OP
440 PRINT "BYTES"
450 PRINT
460 GOTO290
470 SAVE "PET ASCII"
480 RUN

```

THE MACHINE-LANGUAGE PROGRAM

The addresses given here refer to the original location of the ML program in the REM statement. The entry point for program transmission is at label OUTPROG (4176h). The computer is placed in FAST mode, and a 10 second silence follows. If required, the receiving computer's text buffer should be opened during this time. The text byte counter (407Bh) is zeroed, and the beginning address of the program is stored at 403Ch. A CONTROL Q is transmitted to signal beginning of transmission. At label NEWSCREEN (4196h) the screen is cleared. A check for BREAK is made, and a check is done to see if the end of the program has been reached. If either condition is true, then the program jumps to EXIT (41C4h). A CONTROL C is sent, signalling end-of-transmission, and the routine returns to BASIC with the byte-count in the BC register pair.

Otherwise, the successive program lines are printed on the screen, and from there are transmitted by a call to the OUTSCREEN subroutine. This routine will stop and display any line exceeding the 704 raster screen size. Such lines should be deleted to enable OUTPROG to transfer the program. The CONTROL characters are sent in case the receiving terminal program requires them for text loading. They may be changed as required by POKEing the needed values at 4192h and 41C5h.

The OUTSCREEN subroutine transmits text printed on the screen up to the display file character counter (DF-CC), which is used as an end marker. At 4101h, the blank spaces after the last screen line are eliminated by adjusting DF-CC to point right after the last printed character. The screen contents are then scanned and converted at OUTCHAR to PET ASCII characters and graphic symbols, with any appropriate RVS ON and RVS OFF characters. When the last screen character is sent, a jump is made to OUTCR which sends a carriage return and exits the subroutine.

The OUTBYTE subroutine (40D8h) sends out each byte (loaded into the E register) through the computer's TV/TAPE port. Each bit is rotated through the carry bit, and the appropriate SPACE or MARK subroutine is called. The text byte counter is incremented before exiting back to the calling routine.

PET ASCII OUTPUT MACHINE LANGUAGE PROGRAM FOR ZX81 BY CAROL F. MILAZZO

TABLE	4082	20 BE BC A2 BB A1 BF AC A6 A8 A2 22 5C 24 3A 3F	
	4092	28 29 3E 3C 3D 2B 2D 2A 2F 3B 2C 2E 36 31 32 33	
	40A2	34 35 36 37 38 39 41 42 43 44 45 46 47 48 49 4A	
	4082	4B 4C 40 4E 4F 50 51 52 53 54 55 56 57 58 59 5A	
MARK	40C2	DBFE	IN A, (FE)
	40C4	1804	JR DELAY
SPACE	40C6	D3FF	OUT (FF),A
	40C8	1800	JR DELAY
DELAY	40CA	01015F	LD BC,5F01 (2400 BAUD RATE CONSTANT)
LOOP	40CD	10FE	DJNZ LOOP
	40CF	0D	DEC C
	40D0	20FB	JRNZ LOOP
	40D2	AF	XOR A
	40D3	C0	RET NZ
	40D4	C0	RET NZ
	40D5	C9	RET
OUTCR	40D6	1E0D	LD E,0D (SEND CR)
OUTBYTE*	40D8	CDC640	CALL SPACE (START BIT)
	40DB	00	NOP
	40DC	00	NOP
	40DD	37	SCF
NEXTBIT	40DE	CB1B	RR E (FETCH EACH DATA BIT)
	40E0	280D	JRZ STOP
	40E2	3006	JRNC SP
	40E4	D0	RET NC
*	40E5	CDC240	CALL MARK
	40E8	18F4	JR NEXTBIT
SP*	40EA	CDC640	CALL SPACE
	40ED	18EF	JR NEXTBIT
STOP	40EF	1E02	LD E,02 (# OF STOP BITS)
STOPBIT*	40F1	CDC240	CALL MARK
	40F4	1D	DEC E
	40F5	20FA	JRNZ STOPBIT
	40F7	ED4B7B40	LD BC,(407B)
	40FB	03	INC BC (INCREMENT BYTE COUNTER)
	40FC	ED437B40	LD (407B),BC
	4100	C9	RET

OUTSCREEN	4101	2A0E40	LD HL,(DF-CC)	
	4104	AF	XOR A	
COLLAPSE	4105	2B	DEC HL	(COLLAPSE LAST SCREEN LINE)
	4106	BE	CP (HL)	
	4107	28FC	JRZ COLLAPSE	
	4109	CB76	BIT 6,(HL)	
	410B	20F8	JRNZ COLLAPSE	
	410D	23	INC HL	
	410E	220E40	LD (DF-CC),HL	
OUTSCR2	4111	1600	LD D,00	(RVS FLAG OFF)
	4113	2A0C40	LD HL,(D-FILE)	
NEXTCHAR	4116	23	INC HL	
	4117	E5	PUSH HL	
	4118	ED4B0E40	LD BC,(DF-CC)	(CHECK FOR END OF SCREEN)
	411C	A7	AND A	
	411D	ED42	SBC HL,BC	
	411F	7C	LD A,H	
	4120	B5	OR L	
	4121	E1	POP HL	
	4122	28BB	JRZ OUTCR	(RETURN IF END)
	4124	7E	LD A,(HL)	(FETCH A CHARACTER)
	4125	FE76	CP 76	
	4127	2007	JRNZ OUTCHAR	
	4129	18EB	JR NEXTCHAR	(1800 TO SEND CR AFTER EACH LINE)
ENDLINE*	412B	CDD640	CALL OUTCR	
	412E	18E6	JR NEXTCHAR	
OUTCHAR	4130	07	RLCA	
	4131	FE16	CP 16	
	4133	3016	JRNC NORM	
	4135	FE06	CP 06	
	4137	3812	JRC NORM	
	4139	FE08	CP 08	
	413B	380C	JRC INV	
	413D	FE0C	CP 0C	
	413F	380A	JRC NORM	
	4141	FE12	CP 12	
	4143	3804	JRC INV	
	4145	FE14	CP 14	
	4147	3802	JRC NORM	
INV	4149	EE01	XOR 01	
NORM	414B	CB3F	SRL A	
	414D	F5	PUSH AF	
	414E	300C	JRNC RVS-OFF	
RVS-ON	4150	15	DEC D	
	4151	2805	JRZ SET-D	
	4153	1E12	LD E,12	(SEND RVS-ON)
*	4155	CDD840	CALL OUTBYTE	
SET-D	4158	1601	LD D,01	
	415A	180A	JR CONT	
RVS-OFF	415C	15	DEC D	
	415D	2005	JRNZ RESET-D	
	415F	1E92	LD E, 92	(SEND RVS-OFF)
*	4161	CDD840	CALL OUTBYTE	
RESET-D	4164	1600	LD D,00	
CONT	4166	F1	POP AF	
	4167	E5	PUSH HL	
	4168	0600	LD B,00	(TRANSLATE ZX81 TO PET ASCII)
	416A	4F	LD C,A	
*	416B	218240	LD HL, TABLE	
	416E	09	ADD HL,BC	
	416F	5E	LD E,(HL)	

```

*      4170      CDD840      CALL OUTBYTE
      4173      E1          POP HL
      4174      18A0        JR NEXTCHAR
OUTPROG 4176      CDE702      CALL FAST
      4179      210024      LD HL,2400
DELAY1  417C      CD460F      CALL BREAK-1      (CHECK FOR BREAK KEY)
      417F      D0          RET NC              (EXIT IF BREAK PRESSED)
DELAY2  4180      10FE      DJNZ DELAY2
      4182      48          LD C,B
      4183      2B          DEC HL
      4184      7C          LD A,H
      4185      B5          OR L
      4186      20F4        JRNZ DELAY1
      4188      227B40      LD (407B),HL      (ZERO BYTE COUNTER)
      418B      217D40      LD HL,PROGRAM     (START OF PROGRAM)
      418E      22EC40      LD (403C),HL
      4191      1E11        LD E,11          (SEND CONTROL Q)
*      4193      CDD840      CALL OUTBYTE
NEWSCREEN4196    CD2A0A      CALL CLS          (CLEAR SCREEN)
      4199      CD460F      CALL BREAK-1      (CHECK BREAK KEY)
      419C      3026        JRNC EXIT         (EXIT IF BREAK PRESSED)
      419E      2A3C40      LD HL,(403C)      (CHECK FOR END OF PROGRAM)
      41A1      ED4B0C40    LD BC,(D-FILE)
      41A5      A7          AND A
      41A6      ED42        SBC HL,BC
      41A8      7C          LD A,H
      41A9      B5          OR L
      41AA      2818        JRZ EXIT
      41AC      2A3C40      LD HL,(403C)
      41AF      CD4507      CALL PRLINE       (PRINT NEXT BASIC LINE)
      41B2      2A3C40      LD HL,(403C)
      41B5      23          INC HL
      41B6      23          INC HL
      41B7      4E          LD C,(HL)
      41B8      23          INC HL
      41B9      46          LD B,(HL)
      41BA      23          INC HL
      41BB      09          ADD HL,BC
      41BC      223C40      LD (403C),HL      (STORE ADDRESS OF NEXT LINE)
*      41BF      CD0141      CALL OUTSCREEN    (SEND SCREEN)
      41C2      18D2        JR NEWSCREEN
EXIT    41C4      1E03      LD E,03          (SEND CONTROL C)
*      41C6      C3D840      JP OUTBYTE

```

* THESE INSTRUCTIONS REQUIRE OFFSET CALCULATIONS WHEN RELOCATED

CHANGE THE FOLLOWING LINES TO GET STANDARD ASCII OUTPUT FOR OTHER COMPUTERS
BY ELIMINATING THE REVERSE AND GRAPHIC CHARACTERS:

```

TABLE  4082      20 20 20 20 20 20 20 20 20 20 20 22 5C 24 3A 3F
OUTCHAR 4130      E63F          AND 3F
      4132      1833          JR CONT+1

```

TRANSFERRING OTHER DATA

After transferring a program, any arrays or other values in the variables area may also be transmitted by printing them, using BASIC statements, then using RAND USR OUTSCREEN. For example, a 100-element numeric array A(100) can be sent by entering the following lines, and executing GOTO 1:

```
1 FAST
2 FOR I=1 TO 2000
3 NEXT I
4 LET OUTSCREEN=32568 (ML above 32441)
5 LET AD=9000
6 GOSUB 21
7 PRINT " DIM A(100):FOR I=1 TO 100:READ A:NEXT I"
8 GOSUB 18
9 FOR I=0 TO 95 STEP 5
10 PRINT " DATA ";
11 FOR J=1 TO 5
12 PRINT A(I+J);
13 IF J<5 THEN PRINT ".";
14 NEXT J
15 GOSUB 18
16 NEXT I
17 STOP
18 RAND USR OUTSCREEN
19 LET AD=AD+10
20 CLS
21 PRINT AD;
22 RETURN
```

Graphic displays can also be sent by using the OUTSCREEN subroutine. OUTSCREEN normally ignores the NEWLINE character at the end of each display line, but for sending graphic displays these should be used to send a carriage return (CR). This is done by POKEing 00 at 412Fh. Displays produced using the AT command may set DF-CC before the end of the screen text, causing OUTSCREEN to send only that part up to DF-CC. This can be corrected by inserting the command PRINT AT 21,31; before the RAND USR OUTSCREEN within the program. Non-ASCII text, e.g. machine-language files, can be transmitted by using the OUTCR subroutine. Each text byte must be POKEd into 40D7h, followed by executing RAND USR OUTCR. For example:

```
1 FAST
2 FOR I=1 TO 2000
3 NEXT I
4 LET OUTCR=32525 (ML above 32441)
5 FOR I=32441 TO 32767 (Send PET ASCII program)
6 POKE OUTCR+1,PEEK I
7 RAND USR OUTCR
8 NEXT I
```

THE INTERFACE

The hardware interface for the C64 is shown in Figure 1, and consists of the following components:

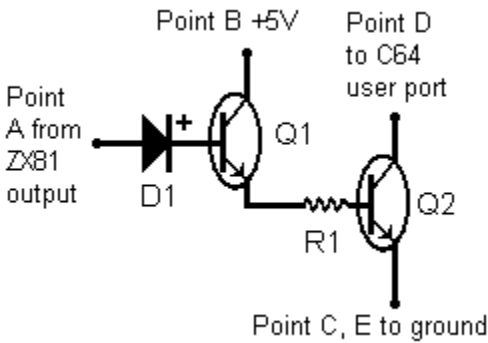


Figure 1. Schematic diagram of hardware interface to connect the output of the Timex/Sinclair ZX81 to a Commodore 64 computer.

PART	RADIO SHACK#	DESCRIPTION
D1	276-1122	1N914 diode
Q1, Q2	276-2009	NPN silicon transistor, 2N2222 etc.
R1	271-010	68 ohm 1/2W resistor
SO1	C1-12	24-contact polarizing connector (DIGI KEY)
	CR1164-1	2 polarizing keys for SO1 (DIGI KEY)

Point "A" is connected to the TV/TAPE output of the computer logic chip. This is pin 16 of IC1 (ULA) on the ZX81/TS1000 (junction of D9 and R29), or pin 53 of U5 (SCLD) on the T51500 (junction of D19 and R20). Point "B" is connected to +5V, and point "C" is connected to ground. In the interface circuit, D1 is used to protect the computer logic chip in case of Q1 failure. Q1 is connected as an emitter follower, identical to the common video output modification for these machines. Q2 converts the signal from Q1 to provide the proper on-off switching required by the Commodore user port. The edgeboard connector is used to connect the interface to the Commodore 64 user port. Point D is connected to pins B and C, and point E is connected to pin A of the user port.

OPERATION

In typical operation, the receiving computer is connected and set to the desired data speed and 8 bit word length. The Sinclair ML program is set up above RAMTOP as described, or in CMOS RAM or other available memory. The desired BASIC program is loaded into the Sinclair/Timex in the usual manner. The command PRINT USR [OUTPROG address] is executed, and the text buffer of the receiving computer is opened. The actual ASCII data transmission is seen as varying horizontal bars on the Sinclair monitor screen. Transmission may be aborted at any time by holding the BREAK key.

After any text transmission is terminated, the number of bytes sent is displayed in the upper left corner of the screen. The text buffer in the receiving computer should be closed, and the transmitted text saved to disk or tape. This ASCII file must be converted into BASIC tokens. The PLUS/TERM program described above includes a utility called "TOKENIZER", which performs this task for disk-drive users. Another utility program called "RECRUNCHER" appeared in July 1985 "Compute!" and works with tape or disk drive. Sinclair BASIC programs will rarely run on other computers without some translation of commands. Translation can be done line-by-line, or with the aid of a utility like "METABASIC" in April 1985 "Compute Gazette."

CONCLUSION

This project is a worthwhile investment, and has saved me the hundreds of thousands of keystrokes that the transfer of an extensive collection of programs would have required. I would appreciate any suggestions for improvements, or other feedback from users. (Send them directly to me, or care of this magazine.) I will supply this program on cassette to anyone who sends a nominal \$5 for the tape and postage.

FIGURE 2

Here is an alternate interface circuit, which provides an RS232-compatible output signal. It uses an LM311 comparator, which is inexpensive, capable of about 100 mA output drive, and is commonly available. However, you could use other comparators or op amps instead.

Power can be provided with a bi-polar 9V supply, or from two 9V "transistor" batteries. Depending on the output drive requirement, a pair of alkaline batteries can run for up to about 28 hours. Note that the power on/off switch must be double-pole, to insure that both batteries are disconnected.

The connection shown is for a non-inverting configuration. If your system requires the signal to be inverted, exchange the two connections marked with "X".

Resistor R3 provides hysteresis (positive feedback) for rapid output transistions. Bypass capacitors C1 and C2 are for glitch-protection, and should be tantalum types rated at 10V or better. Exact capacitance values are non-critical, anywhere from .47 uF to 10 uF. is fine.